SPi | Journal Code: MMA | Article ID 4071 | Dispatch 25.06.16 | CE: Ramon Joseph B. Bato | No. of Pages: 10 | ME:

Research Article

Mathematical Methods in the Applied Sciences

Received 23 May 2016 Published online in Wiley Online Library

(wileyonlinelibrary.com) DOI: 10.1002/mma.4071
MOS subject classification: 34N05; 41-04; 41A60

Modified poisson series to implement the method of multiple scales

Juan F. Navarro[*†]

Communicated by J. Vigo-Aguiar

The method of multiple scales is a global perturbation technique that has resulted to be very useful in perturbed ordinary differential equations characterized by disparate time scales. The general principle behind the method is that the solution to the differential equation is uniformly expanded in terms of two or more independent variables, referred to as time scales. In this article, we present a mathematical object based on a Poisson series to apply the method of multiple scales via specific symbolic computation.

Keywords: symbolic computation; poisson series; perturbation methods; method of multiple scales

1. Introduction

Perturbation theories for differential equations containing a small parameter ϵ are quite old. The small perturbation theory originated by Newton has been highly developed by many others, and an extension of this theory to the asymptotic expansion, consisting on a power series expansion in the small parameter, was devised by Poincaré [1]. The main point is that for the most of the differential equations, it is not possible to obtain an exact solution. In cases where equations contain a small parameter, we can consider it as a perturbation parameter to obtain an asymptotic expansion of the solution. These classical perturbation methods based on asymptotic expansions generally break down because of resonances that lead to what are called secular terms. Multiple scales analysis [2] comprises techniques used to construct uniformly valid approximations to the solutions of perturbation problems, both for small as well as large values of the independent variables. These techniques are useful in systems characterized by disparate time scales, such as weak dissipation in an oscillator. In the standard approach, the expansion of the solution to the perturbed equation depends explicitly on t, ϵt, $\epsilon^2 t$, and the small parameter ϵ itself. The method of multiple scales has become very popular, and it has been applied to a wide range of problems. To cite some examples, Cole and Kevorkian [3], Nayfeh [4–6], Musa [7] and Reiss [8] applied the method of multiple scales to the analysis of weakly linear and nonlinear vibrations modeled by second and third order ordinary differential equations. Rammath and Sandri [9] used it to study equations with variable coefficients. More recently, Krämer [10] has computed an approximate solution to the nonlinear Klein–Gordon equation via the method of multiple scales. Abbasi *et al*. [11] employ the method of multiple scales to analyze the chaotic behavior and different types of fixed points in ferroresonance of voltage transformers considering core loss. This phenomenon has nonlinear chaotic dynamics and includes subharmonic, quasi-periodic, and also chaotic oscillations. In [12] and [13], the authors generalise a computer implementation of the multiple scales method and its application to nonlinear vibration problems. The necessary macro-steps that are used for the development of the computational system are formulated, and the practical ways of encoding these steps using Mathematica are discussed. Cartmell *et al*. [14] investigate the application of the method to some problems in the area of machine and structural dynamics.

In practice, the work involved in the application of this approach to compute the solution to a differential equation cannot be performed by hand, and algebraic processors result to be a very useful tool. Since the early sixties, many systems for specific or general symbolic computation have been developed. Nowadays, many general purpose computer algebra packages contain tools for the calculation of the solution of certain classes of ordinary differential equations. All these packages have the advantage of being very general, so they can deal with a lot of problems of different nature. However, the most common perturbation methods tend to produce expressions containing thousands of terms, and their treatment with those general processors becomes a time-consuming task. Specific symbolic computation packages avoid this inconvenience working with simple data structures and algorithms.

Departamento de Matemática Aplicada, Universidad de Alicante, Carretera San Vicente del Raspeig s/n, 03690 San Vicente del Raspeig, Alicante, Spain
** Correspondence to: Juan F. Navarro, Departamento de Matemática Aplicada, Universidad de Alicante, Carretera San Vicente del Raspeig s/n, 03690 San Vicente del Raspeig, Alicante, Spain.*
† E-mail: jf.navarro@ua.es

The aim of this paper is to propose the mathematical object to implement a general symbolic algorithm to apply the method of multiple scales as presented in the next section.

2. The method of multiple scales

We will consider the initial value problem defined by the following nonlinear second order differential equation:

$$\ddot{x} + x = \epsilon f(x, \dot{x}) , \tag{1}$$

where $0 < \epsilon \ll 1$ is a small parameter and $f(x, \dot{x})$ can be arranged as follows,

$$f(x, \dot{x}) = \sum_{q=0}^{M} \sum_{0 \le \nu \le q} f_{\nu, q-\nu} \, x^{\nu} \, \dot{x}^{q-\nu} , \tag{2}$$

being $f_{\nu,q} \in \mathbb{R}$ for $0 \le \nu, q \le M$, and $M \in \mathbb{N}$. The standard approach [15] is to try a power series solution of the form

$$x(t, \epsilon) = x_0(t) + \epsilon x_1(t) + \epsilon^2 x_2(t) + \cdots .$$

This series is inserted into the governing equation and initial conditions, and coefficients of same powers of ϵ are then grouped to obtain a collection of equations for the coefficient functions $x_i(t)$, which are then solved in a sequential manner. The resulting solution $x(t, \epsilon)$ can be useful in approximating the function when ϵ is small, but it results to be a poor approximation when t is as large as ϵ^{-1}. To determine an expansion valid for times as large as ϵ^{-1}, the combination ϵt should be considered a single variable $T_1 = \epsilon t$. Thus, the truncated expansion is valid for times as large as ϵ^{-1}, but it is not satisfactory when t is as large as ϵ^{-2}. To obtain an adequate asymptotic expansion valid for $t = O(\epsilon^{-2})$, $\epsilon^2 t$ should be considered a single variable $T_2 = \epsilon^2 t$.

The aforementioned discussion suggests that $x(t, \epsilon)$ depends explicitly on $T_0 = t$, $T_1 = \epsilon t$, $T_2 = \epsilon^2 t$, and so on. In order to get a truncated expansion valid for any t up to $O(\epsilon^n)$, we must calculate the dependence of x on the $n + 1$ different time scales $T_0, T_1, \ldots, T_n$. The method of multiple scales considers the expansion to be a function of multiple independent variables, or scales, instead of a single variable t,

$$x(t) = \sum_{\nu=0}^{n-1} \epsilon^{\nu} x_{\nu}(T_0, T_1, \ldots, T_n) + O(\epsilon T_n) , \tag{3}$$

where the independent variables are defined as

$$T_{\nu} = \epsilon^{\nu} t , \quad \nu = 0, 1, 2, \ldots, n . \tag{4}$$

The number of independent time scales depends on the order to which the expansion is carried out. Substitution of Equation (3) into the governing differential equation and collecting coefficients of equal powers of ϵ generate a system of $n + 1$ differential equations. To obtain a uniform solution, the system of differential equations must be solved sequentially for $\nu = 0, 1, \ldots, n - 1$, eliminating secular terms.

The time derivative is transformed according to

$$\frac{d}{dt} = \frac{\partial}{\partial T_0} + \epsilon \frac{\partial}{\partial T_1} + \epsilon^2 \frac{\partial}{\partial T_2} + \cdots . \tag{5}$$

For our purpose this time, we will focus our attention in the functions x_{ν} depending only on two time scales, $T_0 = t$ and $T_1 = \epsilon t$. Of course we can extend this procedure to as many time scales as we like, but because some upcoming equations would become unnecessarily difficult without any further insight into the method itself, we leave it at two time scales. Thus, considering two time scales, we get

$$x(t, \epsilon) = x_0(T_0, T_1) + \epsilon x_1(T_0, T_1) + O(\epsilon^2) .$$

Taking into account the transformation of the time derivative,

$$\frac{d}{dt} = \frac{\partial}{\partial T_0} + \epsilon \frac{\partial}{\partial T_1} ,$$

we get

$$\dot{x} = \frac{\partial}{\partial T_0} x_0 + \epsilon \frac{\partial}{\partial T_1} x_0 + \epsilon \frac{\partial}{\partial T_0} x_1 + O(\epsilon^2) .$$

Let us define the operators

$$D_0 = \frac{\partial}{\partial T_0} , \qquad D_1 = \frac{\partial}{\partial T_1} .$$

Then, equations for $\dot{x}$ and $\ddot{x}$ can be written as

$$\dot{x} = D_0 x_0 + \epsilon \left(D_1 x_0 + D_0 x_1\right) + O(\epsilon^2)\,,$$
$$\ddot{x} = D_0 D_0 x_0 + \epsilon \left(2 D_1 D_0 x_0 + D_0 D_0 x_1\right) + O(\epsilon^2)\,.$$

If we substitute these equations into the governing Equation (1), we get

$$D_0 D_0 x_0 + \epsilon \left(2 D_0 D_1 x_0 + D_0 D_0 x_1\right) + x_0 + \epsilon x_1 = \epsilon \sum_{q=0}^{M} \sum_{0 \le \nu \le q} x_0^{\nu} \left(D_0 x_0\right)^{q-\nu}\,.$$

Now, we collect terms in equal powers of ϵ, to generate the following system of equations,

$$O(1) : D_0 D_0 x_0 + x_0 = 0\,, \tag{6}$$

$$O(\epsilon) : D_0 D_0 x_1 + x_1 = \sum_{q=0}^{M} \sum_{0 \le \nu \le q} x_0^{\nu} \left(D_0 x_0\right)^{q-\nu} - 2 D_0 D_1 x_0\,. \tag{7}$$

The solution to (1) is constructed from the order zero, which corresponds with the unperturbed problem, and can be written as

$$D_0 D_0 x_0 + x_0 = 0\,. \tag{8}$$

The solution to Equation (8) is

$$x_0(T_0, T_1) = R(T_1) \cos\left(T_0 + \Phi(T_1)\right)\,, \tag{9}$$

where $R(T_1)$ and $\Phi(T_1)$ are the slow-varying amplitude and phase of x_0, respectively. Now, we need to compute $D_0 x_0$ and $D_1 D_0 x_0$ in order to obtain x_1. Derivating Equation (9), we get

$$D_0 x_0 = -R(T_1) \sin\left(T_0 + \Phi(T_1)\right)\,, \tag{10}$$

and

$$D_1 D_0 x_0 = -D_1 R(T_1) \sin\left(T_0 + \Phi(T_1)\right) - R(T_1) D_1 \Phi(T_1) \cos\left(T_0 + \Phi(T_1)\right)\,. \tag{11}$$

Now, we can substitute Equations (10) and (11) into Equation (7). First, we need to calculate the products

$$\left(x_0\right)^{\nu} = R^{\nu}(T_1) \cos^{\nu}\left(T_0 + \Phi(T_1)\right)\,,$$
$$\left(D_0 x_0\right)^{q-\nu} = (-1)^{q-\nu} R^{q-\nu}(T_1) \sin^{q-\nu}\left(T_0 + \Phi(T_1)\right)\,,$$

but also

$$-2 D_1 D_0 x_0 = 2 D_1 R(T_1) \sin\left(T_0 + \Phi(T_1)\right) + 2 R(T_1) D_1 \Phi(T_1) \cos\left(T_0 + \Phi(T_1)\right)\,.$$

Thus, Equation (7) can be arranged as follows,

$$D_0 D_0 x_1 + x_1 = \sum_{q=0}^{M} \sum_{0 \le \nu \le q} f_{\nu, q-\nu} (-1)^{q-\nu} R^{\nu}(T_1) \cos^{\nu}\left(T_0 + \Phi(T_1)\right) \sin^{q-\nu}\left(T_0 + \Phi(T_1)\right) +$$
$$+ 2 D_1 R(T_1) \sin\left(T_0 + \Phi(T_1)\right) + 2 R(T_1) D_1 \Phi(T_1) \cos\left(T_0 + \Phi(T_1)\right)\,. \tag{12}$$

Now, we must collect terms is same frequencies in order to avoid the generation of secular terms, by making resonant terms vanish. Q1
To do that, we must expand the right-hand side of Equation (12) by means of the relations

$$\sin(\alpha_1 + \alpha_2) = \sin\alpha_1 \cos\alpha_2 + \sin\alpha_2 \cos\alpha_1\,,$$
$$\cos(\alpha_1 + \alpha_2) = \cos\alpha_1 \cos\alpha_2 - \sin\alpha_1 \sin\alpha_2\,.$$

In the following section, we introduce an example to show clearly the type of expressions we have to handle in order to construct a symbolic computation system to apply the method of multiple scales.

3. Application of the method to the van der Pol equation

In this section, we apply the two timing method to show the mathematical object we have to handle with the specific symbolic computation system. To that purpose, let us consider the van der Pol equation,

$$\ddot{x} + x = \epsilon(1 - x^2)\dot{x}\,. \tag{13}$$

The van der Pol oscillator is a non-conservative oscillator with nonlinear damping. Energy is dissipated at high amplitudes and generated at low amplitudes. Thus, there exists oscillations around a state at which energy generation and dissipation balance. The state towards which the oscillations converge is known as a limit cycle. Van der Pol discovered that no matter the initial conditions, the oscillator converges to a limit cycle. For $\epsilon \ll 1$ and trajectories close to the origin, the amplitude of the oscillation grows very slowly, each oscillation with a different amplitude and period. This behavior gives rise to the concept of multiple timescale of oscillation. The van der Pol oscillator has become the cornerstone for studying systems with limit cycle oscillations in physics, biology, sociology, and even economics.

The $O(1)$ and $O(\epsilon)$ equations ((6) and (7), respectively) become

$$D_0 D_0 x_0 + x_0 = 0\,, \tag{14}$$

and

$$D_0 D_0 x_1 + x_1 = -x_0^2 D_0 x_0 + D_0 x_0 - 2D_0 D_1 x_0\,. \tag{15}$$

The solution to (14) is

$$x_0(T_0, T_1) = R(T_1)\cos(T_0 + \Phi(T_1))\,,$$

where $R(T_1)$ and $\Phi(T_1)$ are the slow-varying amplitude and phase of the oscillation. The derivatives $D_0 x_0$ and $D_1 D_0 x_0$ can be written as

$$D_0 x_0 = -R(T_1)\sin(T_0 + \Phi(T_1))$$

and

$$D_1 D_0 x_0 = -D_1 R(T_1)\sin(T_0 + \Phi(T_1)) - R(T_1) D_1\Phi(T_1)\cos(T_0 + \Phi(T_1))\,.$$

Now, we substitute the solution to (14) as well as its derivatives into (15) to get

$$\begin{aligned} D_0 D_0 x_1 + x_1 = {} & R^3(T_1)\cos^2(T_0 + \Phi(T_1))\sin(T_0 + \Phi(T_1)) - R(T_1)\sin(T_0 + \Phi(T_1)) + \\ & + 2D_1 R(T_1)\sin(T_0 + \Phi(T_1)) + 2R(T_1) D_1\Phi(T_1)\cos(T_0 + \Phi(T_1))\,. \end{aligned} \tag{16}$$

Taking into account the relation

$$\cos^2(T_0 + \Phi(T_1))\sin(T_0 + \Phi(T_1)) = \frac{1}{4}\sin(T_0 + \Phi(T_1)) + \frac{1}{4}\sin(3T_0 + 3\Phi(T_1))\,,$$

we can expand Equation (16) to obtain

$$\begin{aligned} D_0 D_0 x_1 + x_1 = {} & \frac{1}{4}R^3(T_1)\sin(T_0 + \Phi(T_1)) + \frac{1}{4}R^3(T_1)\sin(3T_0 + 3\Phi(T_1)) - \\ & - R(T_1)\sin(T_0 + \Phi(T_1)) + 2D_1 R(T_1)\sin(T_0 + \Phi(T_1)) + 2R(T_1) D_1\Phi(T_1)\cos(T_0 + \Phi(T_1))\,. \end{aligned} \tag{17}$$

Now, we can collect terms is same frequencies in order to avoid the generation of secular terms,

$$\begin{aligned} D_0 D_0 x_1 + x_1 = {} & \left(\frac{1}{4}R^3(T_1) - R(T_1) + 2D_1 R(T_1)\right)\sin(T_0 + \Phi(T_1)) + \\ & + 2R(T_1) D_1\Phi(T_1)\cos(T_0 + \Phi(T_1)) + \frac{1}{4}R^3(T_1)\sin(3T_0 + 3\Phi(T_1))\,. \end{aligned} \tag{18}$$

Thus, resonant terms vanish if we make

$$\begin{aligned} H_1(T_1) &= \frac{1}{4}R^3(T_1) - R(T_1) + 2D_1 R(T_1) = 0\,, \\ H_2(T_1) &= 2R(T_1) D_1\Phi(T_1) = 0\,. \end{aligned}$$

So the conditions to avoid secular terms take the form of two differential equations,

$$\begin{aligned} 2D_1 R(T_1) &= R(T_1) - \frac{1}{4}R^3(T_1) - R(T_1) + 2D_1 R(T_1), \\ D_1\Phi(T_1) &= 0\,. \end{aligned}$$

The right-hand side of both equations is a polynomial in R and Φ, so we can compute the zeros of both sides. In this case, we get that $R(T_1) = 0, \pm 2$ and $\Phi(T_1)$ is a linear function of T_1. A numerical exploration is enough to show that $R(T_1) = 2$ is a stable fixed point of the corresponding differential equation.

4. Mathematical object

From the analysis of Section 3, we see that a modification of a Poisson series is enough to handle the expressions involved in the application of the method of multiple scales. In this section, we will study the way the structure of a Poisson series must be adapted in order to implement the method of multiple scales as a symbolic algorithm.

A Poisson series is an expression of the form

$$S = \sum_{i \in \mathbb{I}} \sum_{j \in \mathbb{K}} C_{i,j} A_1^{i_1} A_2^{i_2} \cdots A_m^{i_m} \left\{ \begin{matrix} \cos \\ \sin \end{matrix} \right\} (j_1 \Phi_1 + \cdots + j_n \Phi_n) . \tag{19}$$

The summations bear on collections $\mathbb{I}$ and $\mathbb{K}$ consisting of vector of integers $i = (i_1, i_2, \ldots, i_m)$ and $j = (j_1, j_2, \ldots, j_n)$, the coefficients $C_{i,j}$ belongs to $\mathbb{R}$. The term $A_1^{i_1} A_2^{i_2} \cdots A_m^{i_m}$ is usually referred to as monomial, and the letters $A_1, A_2, \ldots, A_m$ as the polynomial variables. The greek letters $\Phi_1, \Phi_2, \ldots, \Phi_n$ constitute the angle variables.

The set of Poisson series forms a commutative algebra over the ring of coefficients [16]. If P and Q are Poisson series, then their sum and product by a real number, $P + Q$ and αP, with $\alpha \in \mathbb{R}$, are also Poisson series. Algebraic closure properties make automatic manipulation rather easy when the elements in the algebra are represented in a standard canonical form, because closure implies that the result retains the standard form of the operands.

Now, we will proceed to introduce some elements in the Poisson series structure at the computational level in order to be able to apply the method of multiple scales. Let $\mathbb{T}$ be the set of independent variables (time scales) $T_0, T_1, \ldots, T_n$, that is,

$$\mathbb{T} = \{T_0, T_1, \ldots, T_n\} ,$$

and $\mathbb{P}$ be the power set of $\mathbb{T}$, $\mathbb{P} = \mathcal{P}(\mathbb{T})$. For example, if we consider $n = 1$, then $\mathbb{T} = \{T_0, T_1\}$, with $T_0 = t, T_1 = \epsilon t$, and $\mathbb{P} = \{\emptyset, \{T_0\}, \{T_1\}, \{T_0, T_1\}\}$. We will use these sets to establish how the integration constants depend on the different time scales.

Now, let us consider the m variables $A_1(\tau_1), A_2(\tau_2), \ldots, A_m(\tau_m)$, where $\tau_1, \ldots, \tau_m \in \mathbb{P}$. In the example earlier, $n = 1, m = 2, A_1(\tau_1) = R(T_1)$ and $A_2(\tau_2) = \Phi(T_1)$, that is, $\tau_1 = \tau_2 = \{T_1\}$.

In order to apply the method of multiple scales as described in the example of Section 3, we also need to introduce the partial derivative of a variable with respect to a time scale in the symbolic object at the computational level. For that purpose, we first introduce the notation, $\sigma = (\sigma_2, \sigma_1)$, being $\sigma_1, \sigma_2 \in \{-1, 0, 1, \ldots, n\}$. Then, we define the operator

$$D_{\sigma_2, \sigma_1} = \frac{\partial^2}{\partial T_{\sigma_2} \partial T_{\sigma_1}} .$$

If σ_1 or σ_2 equals -1, the corresponding partial derivative is not performed.

In order to clarify how we introduce this operator, let us refer to the example described in Section 3, where $n = 1, A_1(\tau_1) = R(T_1)$ and $A_2(\tau_2) = \Phi(T_1)$. The solution to Equation (14) is written as a symbolical object of the form

$$x_0(T_0, T_1) = A_1(\tau_1) D_{-1,-1} A_1(\tau_1) \cos(T_0 + A_2(\tau_2)) .$$

The operator $D_{-1,-1}$ implies that no partial derivative affects the variable $A_1(\tau_1)$, so we do not have to take into account this part of the term, that is,

$$D_{-1,-1} A_1(\tau_1) = 1 .$$

The derivative of $x_0(T_0, T_1)$ with respect to T_0 is written as

$$D_0 x_0(T_0, T_1) = -A_1(\tau_1) D_{-1,-1} A_1(\tau_1) \sin(T_0 + A_2(\tau_2)) .$$

We see in the equation earlier that no partial derivative acts on $A_1(\tau_1)$, as this variable does not depend on T_0. As before, $D_{-1,-1} A_1(\tau_1) = 1$. Now, the partial derivative of $D_0 x_0(T_0, T_1)$ with respect to T_1 reads

$$D_1 D_0 x_0(T_0, T_1) = -D_{-1,1} A_1(\tau_1) \sin(T_0 + A_2(\tau_2)) - A_1(\tau_1) D_{-1,-1} A_1(\tau_1) D_{-1,1} A_2(\tau_2) \cos(T_0 + A_2(\tau_2)) .$$

Here, $D_{-1,-1} A_1(\tau_1) = 1$, and

$$D_{-1,1} A_1(\tau_1) = \frac{\partial}{\partial T_1} A_1(\tau_1) , \quad D_{-1,1} A_2(\tau_2) = \frac{\partial}{\partial T_1} A_2(\tau_2) .$$

In order to handle the more general type of mathematical expression that appears in the application of the method of multiple scales as described in Sections 2 and 3, we will consider the following modified Poisson series,

$$S = \sum_{\nu \in \mathbb{I}} P_\nu(A_1(\tau_1), \dots, A_m(\tau_m)) \sin(\nu_0 T_0 + \nu_1 A(\tau_1) + \cdots + \nu_m A_m(\tau_m)) + \\ + Q_\nu(A_1(\tau_1), \dots, A_m(\tau_m)) \cos(\nu_0 T_0 + \nu_1 A(\tau_1) + \cdots + \nu_m A_m(\tau_m)), \tag{20}$$

where the summations bear on the collection $\mathbb{I}$ consisting of vectors of integers $\nu = (\nu_0, \nu_1, \nu_2, \dots, \nu_m)$, and P_ν and Q_ν are expressions of the form

$$P_\nu(A_1(\tau_1), \dots, A_m(\tau_m)) = \sum_{i \in \mathbb{K}} C_i A_1^{i_1}(\tau_1) D_{\sigma_{i_1}} A_1(\tau_1) \times A_2^{i_2}(\tau_2) D_{\sigma_{i_2}} A_2(\tau_2) \times \cdots \times A_m^{i_m}(\tau_m) D_{\sigma_{i_m}} A_m(\tau_m), \tag{21}$$

and

$$Q_\nu(A_1(\tau_1), \dots, A_m(\tau_m)) = \sum_{j \in \mathbb{K}} C_j A_1^{j_1}(\tau_1) D_{\sigma_{j_1}} A_1(\tau_1) \times A_2^{j_2}(\tau_2) D_{\sigma_{j_2}} A_2(\tau_2) \times \cdots \times A_m^{j_m}(\tau_m) D_{\sigma_{j_m}} A_m(\tau_m). \tag{22}$$

Here, the summations bear on the collection $\mathbb{K}$ consisting of vectors of integers $\nu = (\nu_1, \dots, \nu_m)$, and $\sigma_{i_\nu} = (\sigma_{i_{\nu,2}}, \sigma_{i_{\nu,1}})$, with $\sigma_{i_{\nu,2}}, \sigma_{i_{\nu,1}} \in \{-1, 0, 1, \dots, n\}$.

The letters $A_1, A_2, \dots, A_m$ are the polynomial and angle variables of the Poisson series. These variables depend on the time scales $T_0, T_1, \dots, T_n$ as given by $\tau_1, \tau_2, \dots, \tau_m \in \mathbb{P}$,

$$A_1 = A_1(\tau_1), \quad A_2 = A_2(\tau_2), \quad \dots \quad A_m = A_m(\tau_m).$$

The operators $D_{\sigma_{i_\nu}} = D_{(\sigma_{i_{\nu,2}}, \sigma_{i_{\nu,1}})}$, defined as

$$D_{(\sigma_{i_{\nu,2}}, \sigma_{i_{\nu,1}})} = \frac{\partial^2}{\partial T_{\sigma_{i_{\nu,2}}} \partial T_{\sigma_{i_{\nu,1}}}},$$

for any $\sigma_{i_{\nu,1}}, \sigma_{i_{\nu,2}} \neq -1$, act on the corresponding variable A_ν, for any $\nu = 1, \dots, m$. These operators only act on the polynomial part of the term. If $\sigma_{i_{\nu,2}} = -1$ and $\sigma_{i_{\nu,1}} \neq -1$, then

$$D_{(\sigma_{i_{\nu,2}}, \sigma_{i_{\nu,1}})} = \frac{\partial}{\partial T_{\sigma_{i_{\nu,1}}}}.$$

On the other hand, if $\sigma_{i_{\nu,1}} = \sigma_{i_{\nu,2}} = -1$, then

$$D_{(\sigma_{i_{\nu,2}}, \sigma_{i_{\nu,1}})}(A_\nu(\tau_\nu)) = 1.$$

Now, we look for a canonical representation for each equivalence class defined in the set of modified Poisson series [16]. For that purpose, the following operations must be performed over each series:

1. Let us consider a modified Poisson series

$$S = \sum_{\nu \in \mathbb{I}} P_\nu(A_1(\tau_1), \dots, A_m(\tau_m)) \sin(\nu_0 T_0 + \nu_1 A(\tau_1) + \cdots + \nu_m A_m(\tau_m)) + \\ + Q_\nu(A_1(\tau_1), \dots, A_m(\tau_m)) \cos(\nu_0 T_0 + \nu_1 A(\tau_1) + \cdots + \nu_m A_m(\tau_m)).$$

   If $\nu_0 < 0$, the following rules must be applied:

$$\sin(\nu_0 T_0 + \alpha) = -\sin(-\nu_0 T_0 - \alpha), \quad \cos(\nu_0 T_0 + \alpha) = \cos(-\nu_0 T_0 - \alpha).$$

   Moreover, terms with identical trigonometric part must be grouped together.
2. The terms of a modified Poisson series must be ordered in the following way. Let us consider two terms of a modified Poisson series,

$$S_\nu = P_\nu(A_1(\tau_1), \dots, A_m(\tau_m)) \sin(\nu_0 T_0 + \nu_1 A(\tau_1) + \cdots + \nu_m A_m(\tau_m)) + \\ + Q_\nu(A_1(\tau_1), \dots, A_m(\tau_m)) \cos(\nu_0 T_0 + \nu_1 A(\tau_1) + \cdots + \nu_m A_m(\tau_m)),$$

   and

$$S_\alpha = P_\alpha(A_1(\tau_1), \dots, A_m(\tau_m)) \sin(\alpha_0 T_0 + \alpha_1 A(\tau_1) + \cdots + \alpha_m A_m(\tau_m)) + \\ + Q_\alpha(A_1(\tau_1), \dots, A_m(\tau_m)) \cos(\alpha_0 T_0 + \alpha_1 A(\tau_1) + \cdots + \alpha_m A_m(\tau_m)).$$

We say that $S_\nu < S_\alpha$ if for the first integer $i \in \mathbb{Z}$, $0 \leq i \leq m$, such that $\nu_i \neq \alpha_i$, then $\nu_i < \alpha_i$.

3. The polynomials P_ν and Q_ν of each term of the modified Poisson series must be ordered as follows. Let

$$P_\nu(A_1(\tau_1),\ldots,A_m(\tau_m)) = \sum_{i\in\mathbb{K}} C_i A_1^{i_1}(\tau_1) D_{\sigma_{i_1}} A_1(\tau_1) \times A_2^{i_2}(\tau_2) D_{\sigma_{i_2}} A_2(\tau_2) \times \cdots \times A_m^{i_m}(\tau_m) D_{\sigma_{i_m}} A_m(\tau_m)\,,$$

be a polynomial part of a term of a modified Poisson series. Here, $i = (i_1,\ldots,i_m) \in \mathbb{K}$, and $\sigma_{i_\nu} = (\sigma_{i_{\nu,2}}, \sigma_{i_{\nu,1}})$, with $\sigma_{i_{\nu,2}}, \sigma_{i_{\nu,1}} \in \{-1, 0, 1, \ldots, n\}$. Let us consider now two terms of this polynomial part,

$$B_i = C_i A_1^{i_1}(\tau_1) D_{\sigma_{i_1}} A_1(\tau_1) A_2^{i_2}(\tau_2) D_{\sigma_{i_2}} A_2(\tau_2) \times \cdots \times A_m^{i_m}(\tau_m) D_{\sigma_{i_m}} A_m(\tau_m)\,,$$

and

$$B_j = C_j A_1^{j_1}(\tau_1) D_{\sigma_{j_1}} A_1(\tau_1) A_2^{j_2}(\tau_2) D_{\sigma_{j_2}} A_2(\tau_2) \times \cdots \times A_m^{j_m}(\tau_m) D_{\sigma_{j_m}} A_m(\tau_m)\,.$$

For the sake of simplicity, let us introduce the vector of integers

$$\varsigma = (\varsigma_1,\ldots,\varsigma_{3m}) \in \Omega\,,$$

such that

$$\varsigma_1 = i_1\,,\quad \varsigma_2 = \sigma_{i_1,2}\,,\quad \varsigma_3 = \sigma_{i_1,1}\,,\quad \varsigma_4 = i_2\,,\quad \varsigma_5 = \sigma_{i_2,2}\,,\quad \varsigma_6 = \sigma_{i_2,1}\,,$$

and so on. In general,

$$\varsigma_{3p+1} = i_{p+1}\,,\quad \varsigma_{3p+2} = \sigma_{i_{p+1},2}\,,\quad \varsigma_{3p+3} = \sigma_{i_{p+1},1}\,, \tag{23}$$

for any integer p such that $0 \leq p < m$. Let Ω be the collection of vectors of integers of the form $\varsigma = (\varsigma_1,\ldots,\varsigma_{3m})$. Let us consider now the corresponding vectors of indices $\varsigma^{(i)}$ and $\varsigma^{(j)}$, associated to terms B_i and B_j, respectively. We say that $B_i < B_j$ if for the first integer k such that $\varsigma_k^{(i)} \neq \varsigma_k^{(j)}$, then $\varsigma_k^{(i)} < \varsigma_k^{(j)}$ is satisfied.

5. Data structure for a modified Poisson series

Now, we will consider the special Poisson series set we are working with from the computational point of view. To that purpose, we will analyze the basic information that characterizes a modified Poisson series, as well as the data structure to store it in the computer. This must be done preserving the canonical representation we have chosen.

First of all, we must set the constants m and n, characterizing the number of variables, $A_1,\ldots,A_m$, and the number of time variables, $T_0, T_1,\ldots,T_n$, respectively. Once these constants are set, let us consider the m variables $A_1(\tau_1), A_2(\tau_2),\ldots,A_m(\tau_m)$, where $\tau_1,\ldots,\tau_m \in \mathbb{P}$.

We need to represent the dependence of each variable A_ν on the set of independent variables $T_0, T_1,\ldots,T_n$ through τ_ν, for any $\nu = 1,\ldots,m$. To that purpose, we will use a $m \times (n+1)$ matrix, M_τ. Each row ν of the matrix indicates the dependence of the corresponding variable A_ν on the time variables. Thus,

$$(M_\tau)_{\nu,\alpha} = \begin{cases} 1 & \text{if } A_\nu \text{ depends on } T_\alpha\,, \\ 0 & \text{otherwise}\,, \end{cases}$$

for any $\nu = 1,\ldots,m$ and $\alpha = 0,\ldots,n$. For example, considering the example introduced in Section 3, with $m = 2, n = 1, A_1(\tau_1) = R(T_1)$ and $A_2(\tau_2) = \Phi(T_1)$, we have the following dependence matrix,

$$M_\tau = \begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix} \begin{matrix} \leftarrow A_1(\tau_1) \\ \leftarrow A_2(\tau_2) \end{matrix}.$$

The efficiency of the algorithms for the basic algebra of a series depends on the way is coded. An overcoded structure that makes good use of memory generally requires complex algorithms, which increase the computational cost in terms of time. On the other hand, an undercoded computational representation of the series generates simple algorithms, because the location of all the coefficients can be obtained directly [17]. However, this scheme presents the inconvenience of being very wasteful in the memory resources required for the storage of the series. As pointed out in [16], most of the operations involving a series are based on navigating and searching through the structure that represents the series.

Let us introduce now the red–black tree structure. A red–black tree is a special type of tree, where each node has a color attribute, the value of which is either red or black. In addition to the ordinary requirements imposed on binary search trees, the following additional requirements of any valid red-black tree apply: A node is either red or black. The root is black. All leaves are black, even when the parent is black. Both children of every red node are black. Every simple path from a node to a descendant leaf contains the same number of black nodes. A critical property of red-black trees is enforced by these constraints: the longest path from the root to a leaf is no more than twice as long as the shortest path from the root to a leaf in that tree. The result is that the tree is roughly balanced. Because operations such as inserting, deleting, and finding values requires worst-case time proportional to the height of the tree, this fact makes the red–black tree be a efficient. For instance, the search–time results to be $O(\log n)$.

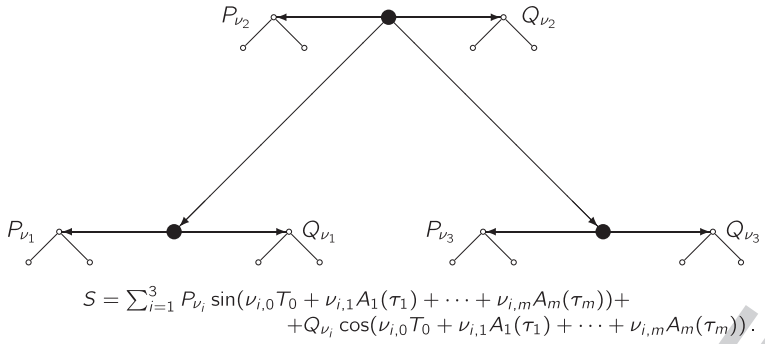


Figure 1. A red–black tree represents a Poisson series. Here, $\nu_1 = (\nu_{1,0}, \nu_{1,1}, \ldots, \nu_{1,m})$, $\nu_2 = (\nu_{2,0}, \nu_{2,1}, \ldots, \nu_{2,m})$ and $\nu_3 = (\nu_{3,0}, \nu_{3,1}, \ldots, \nu_{3,m})$, where $\nu_1 < \nu_2 < \nu_3$.

Thus, the most adequate structure for storing a modified Poisson series is a red–black tree for the storage of the trigonometric part, where each node of the structure stores a key, given by the vector of integers $\nu = (\nu_0, \nu_1, \ldots, \nu_m)$ representing the angle of the trigonometric part. Each of these nodes is linked to two red–black trees for the storage of the polynomials P_ν and Q_ν, respectively.

F1 In Figure 1, we show the structure used to represent the modified Poisson series given by

$$\begin{aligned} S = {} & P_{\nu_1}(A_1(\tau_1), \ldots, A_m(\tau_m)) \sin\left(\nu_{1,0} T_0 + \sum_{i=1}^{m} \nu_{1,i} A_i(\tau_i)\right) + \\ & + Q_{\nu_1}(A_1(\tau_1), \ldots, A_m(\tau_m)) \cos\left(\nu_{1,0} T_0 + \sum_{i=1}^{m} \nu_{1,i} A_i(\tau_i)\right) + \\ & + P_{\nu_2}(A_1(\tau_1), \ldots, A_m(\tau_m)) \sin\left(\nu_{2,0} T_0 + \sum_{i=1}^{m} \nu_{2,i} A_i(\tau_i)\right) + \\ & + Q_{\nu_2}(A_1(\tau_1), \ldots, A_m(\tau_m)) \cos\left(\nu_{2,0} T_0 + \sum_{i=1}^{m} \nu_{2,i} A_i(\tau_i)\right) + \\ & + P_{\nu_3}(A_1(\tau_1), \ldots, A_m(\tau_m)) \sin\left(\nu_{3,0} T_0 + \sum_{i=1}^{m} \nu_{3,i} A_i(\tau_i)\right) + \\ & + Q_{\nu_3}(A_1(\tau_1), \ldots, A_m(\tau_m)) \cos\left(\nu_{3,0} T_0 + \sum_{i=1}^{m} \nu_{3,i} A_i(\tau_i)\right). \end{aligned}$$

In this figure, it is assumed that $\nu_1 < \nu_2 < \nu_3$.

Now, we will concentrate on the way the polynomial parts P_ν and Q_ν are stored. As explained earlier, each node of the Poisson series structure is linked with two red–black trees that represent P_ν and Q_ν, respectively. Let us now consider one of these two polynomials,

$$P_\nu(A_1(\tau_1), \ldots, A_m(\tau_m)) = \sum_{i \in \mathbb{K}} C_i A_1^{i_1}(\tau_1) D_{\sigma_{i_1}} A_1(\tau_1) \times A_2^{i_2}(\tau_2) D_{\sigma_{i_2}} A_2(\tau_2) \times \cdots \times A_m^{i_m}(\tau_m) D_{\sigma_{i_m}} A_m(\tau_m).$$

The information associated to each term of P_ν is given by the following elements:

1. A real number $C_i \in \mathbb{R}$.
2. A set of $3m$ integer numbers, $i_1, \sigma_{i_{1,2}}, \sigma_{i_{1,1}}, \ldots, i_m, \sigma_{i_{m,2}}, \sigma_{i_{m,1}}$.

F2 The data associated to each node of the tree is a real number representing the coefficient of the corresponding term (C_i), and the key of each node is given by the set $i_1, \sigma_{i_{1,2}}, \sigma_{i_{1,1}}, \ldots, i_m, \sigma_{i_{m,2}}, \sigma_{i_{m,1}}$. In Figure 2, we show the tree structure in which a polynomial is stored. The polynomial we show in Figure 2 has seven terms, with keys $\nu_1 < \nu_2 < \cdots < \nu_7$. Each key corresponds to a vector of integer numbers of the form

$$\nu = \left(\nu_1, \sigma_{\nu_{1,2}}, \sigma_{\nu_{1,1}}, \ldots, \nu_m, \sigma_{\nu_{m,2}}, \sigma_{\nu_{m,1}}\right).$$

If we store the key of a term in a vector structure, the complexity of the comparison of the keys is $O(m)$. We can reduce this complexity by storing keys in red–black trees. For each term of a Poisson series, we store vectors $(\nu, i_\nu, \sigma_{i_{\nu,2}}, \sigma_{i_{\nu,1}})$. Here, ν represents the number of the corresponding variable. Thus, the complexity of comparison between terms is reduced from $O(m)$ to $O(\log_2(m))$ in the worst case scenario. If the keys associated to two different terms have different size, that means that both terms are not equal and cannot be collected. This fact helps also to reduce the computation time. Moreover, it is not necessary to compare the entire key in case one index fails.

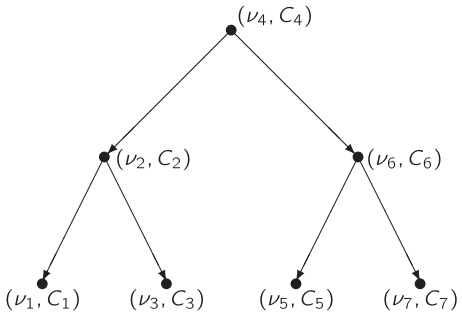


Figure 2. Red–black tree for representing a polynomial with keys $\nu_1 < \nu_2 < \nu_3 < \nu_4 < \nu_5 < \nu_6 < \nu_7$. Each node of the tree contains also the value of the coeffcient of the term (C_i).

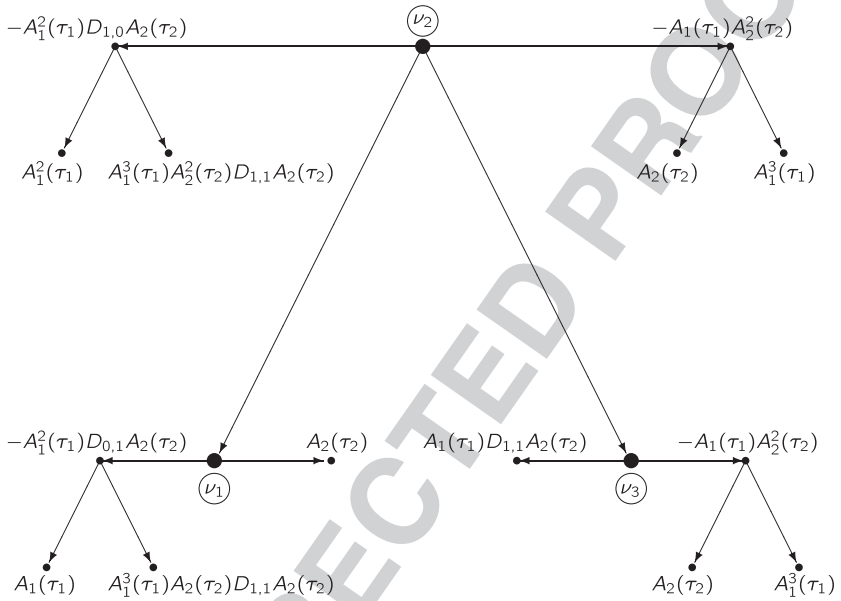


Figure 3. Red–black tree structure for the storage of the modified Poisson series given in equation (24).

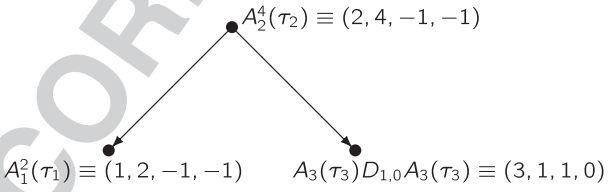


Figure 4. Representation of the key $K = A_1^2(\tau_1)A_2^4(\tau_2)A_3(\tau_3)D_{1,0}A_3(\tau_3)$ in a red–black tree.

Thus, from a computational point of view, a modified Poisson series will be represented by a red–black tree where each node is linked to two red–black trees with keys stored in red–black trees. In Figure 3, we show the representation of the Poisson series **F3**

$$\begin{aligned} S = {} & \left(A_1(\tau_1) - A_1^2(\tau_1)D_{0,1}A_2(\tau_2) + A_1^3(\tau_1)A_2(\tau_2)D_{1,1}A_2(\tau_2)\right)\sin(T_0 + A_1(\tau_1)) - \\ & + (A_2(\tau_2))\cos(T_0 + A_1(\tau_1)) + \\ & + \left(A_1^2(\tau_1) - A_1^2(\tau_1)D_{1,0}A_2(\tau_2) + A_1^3(\tau_1)A_2^2(\tau_2)D_{1,1}A_2(\tau_2)\right)\sin(T_0 + 2A_1(\tau_1)) + \\ & + \left(A_2(\tau_2) - A_1(\tau_1)A_2^2(\tau_2) + A_1^3(\tau_1)\right)\cos(T_0 + 2A_1(\tau_1)) + \\ & + (A_1(\tau_1)D_{1,1}A_2(\tau_2))\sin(3T_0 + A_1(\tau_1) - A_2(\tau_2)) + \\ & + \left(A_2(\tau_2) - A_1(\tau_1)A_2^2(\tau_2) + A_1^3(\tau_1)\right)\cos(3T_0 + A_1(\tau_1) - A_2(\tau_2)), \end{aligned} \tag{24}$$

just to clarify the way red–black trees are used to store a Poisson series.

We also illustrate the way the key is coded in a red–black tree structure. In Figure 4, we show the representation of the key **F4**

$$K = A_1^2(\tau_1)A_2^4(\tau_2)A_3(\tau_3)D_{1,0}A_3(\tau_3)\,,$$

considering that $m = 3$. It is convenient to say that we need to store the complete key in the tree structure, given by

$$K = A_1^2(\tau_1)D_{-1,-1}A_1(\tau_1)A_2^4(\tau_2)D_{-1,-1}A_2(\tau_2)A_3(\tau_3)D_{1,0}A_3(\tau_3)\,.$$

Here, we have the following vectors to be stored in the red–black tree structure: $(1, 2, -1, -1)$, $(2, 4, -1, -1)$, and $(3, 1, 1, 0)$.

6. Conclusions

In this paper, we analyze the method of multiple scales from the point of view of the symbolic computation to find the mathematical object that is needed to implement a general algorithm to apply the method of multiple scales in its standard version. As a consequence of the analysis of the method, we see that a modification of a Poisson series is enough to handle the expressions involved in the application of the method of multiple scales. In Section 4, we study the way the structure of a Poisson series must be adapted in order to implement the method as a symbolic algorithm. Finally, we deal with the most adequate structure for storing these series, a red–black tree for the storage of the trigonometric part, where each node of the structure stores a key, given by the vector of integers representing the angle of the trigonometric part. Each of these nodes is linked to two red–black trees for the storage of the polynomials P_ν and Q_ν, respectively. Moreover, we propose storing the key of a term in a vector structure, so the complexity of the comparison is reduced from $O(m)$ to $O(\log_2 m)$ in the worst case scenario.

References

1. Poincaré H. *Les Methodes Nouvelles de la Mécanique Celeste I*. Gauthiers-Villars, 1892.
2. Nayfeh AH. *Introduction to Perturbation Techniques*. Wiley & Sons: New York, 1981.
3. Cole JD, Kevorkian J. Uniformly valid asymptotic approximations for certain nonlinear differential equations. *Nonlinear Differential Equations and Nonlinear Mechanics*. Academic: New York, 1963, 113–120.
4. Nayfeh AH. A perturbation method for treating nonlinear oscillation problems. *Journal of Mathematical Physics* 1965; **44**:368–374.
5. Nayfeh AH. The van der Pol oscillator with delayed amplitude limiting. *Proceedings of the IEEE 55*, 1967, 111–
6. Nayfeh AH. Forced oscillations of the van der Pol oscillator with delayed amplitude limiting. *IEEE Transactions on Circuit Theory* 1968; **15**:192–200.
7. Musa SA. Integral constraints in weakly nonlinear periodic systems. *SIAM Journal on Applied Mathematics* 1967; **15**:1324–13
8. Reiss EL. On multivariable asymptotic expansions. *SIAM Review* 1971; **13**:189–196.
9. Ramnath RV, Sandri G. A generalized multiple scales approach to a class of linear differential equations. *Journal of Mathematical Analysis and Applications* 1969; **28**:339–364.
10. Krämer P. The Method of Multiple Scales for nonlinear Klein–Gordon and Schrödinger Equations, *Doctoral disserttion*, 2013.
11. Abbasi A, Fathi SH, Gharehpatian GB, Gholami A, Abbasi HR. Voltage transformer ferroresonance analysis using multiple scales method and chaos theory. *Complexity* 2013; **18**(6):34–45.
12. Khanin R, Cartmell MP, Gilbert A. Applying the perturbation method of multiple scales. *Mathematica in Education and Research* 1999; **8**(2):19–26.
13. Khanin R, Cartmell MP, Gilbert A. A computerised implementation of the multiple scales perturbation method using mathematica. *Computers and Structures* 2000; **76**:565–575.
14. Cartmell MP, Ziegler SW, Khanin R, Forehand DIM. Multiple scales analyses of the dynamics of weakly nonlinear mechanical systems. *Transactions of the ASME, Applied Mechanics Reviews* 2003; **56**(5):455–492.
15. Minorsky N. *Nonlinear oscillations*. Krieger: Florida, 1987.
16. San–Juan F, Abad A. Algebraic and symbolic manipulation of poisson series. *Journal of Symbolic Computation* 2001; **32**:565–572.
17. Henrard J. A survey of Poisson series processors. *Celestial Mechanics and Dynamical Astronomy* 1989; **45**:245–253.

Author Query Form

Journal: Mathematical Methods in the Applied Sciences

Article: MMA_4071

Dear Author,

During the copyediting of your paper, the following queries arose. Please respond to these by annotating your proof with the necessary changes/additions.

- If you intend to annotate your proof electronically, please refer to the E-annotation guidelines.
- If you intend to annotate your proof by means of hard-copy mark-up, please use the standard proof-reading marks in annotating corrections. If manually writing corrections on your proof and returning it by fax, do not write too close to the edge of the paper. Please remember that illegible mark-ups may delay publication.

Whether you opt for hard-copy or electronic annotation of your proof, we recommend that you provide additional clarification of answers to queries by entering your answers on the query sheet, in addition to the text mark-up.

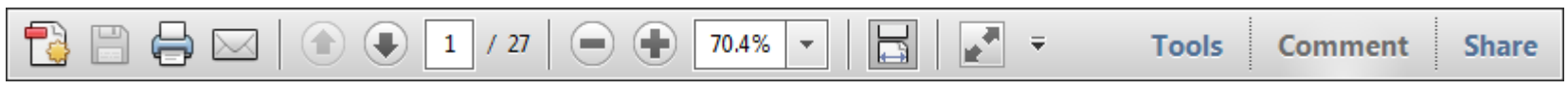
Query No.	Query	Remark
Q1	AUTHOR: Now, we must collect terms is same...order to avoid the generation of secular. The meaning of this sentence is not clear; please rewrite or confirm that the sentence is correct.	
Q2	AUTHOR: Please provide the city location of publisher for Reference 1.	
Q3	AUTHOR: “J. Math. and Phys.” has been changed to “Journal of Mathematical Physics”. Please check.	
Q4	AUTHOR: Please provide city location where the conference was held for Reference 5.	
Q5	AUTHOR: “lEEE Trand. Circuit Theory” has been changed to IEEE Transactions on Circuit Theory”. Please check.	
Q6	AUTHOR: “Celest. Mech.” has been changed to “Celestial Mechanics and Dynamical Astronomy”. Please check.	

WILEY

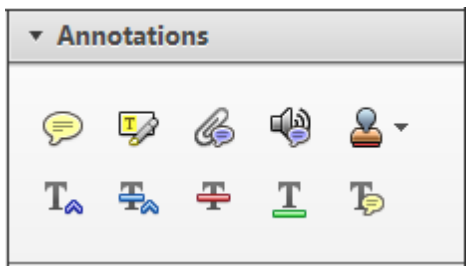
USING e-ANNOTATION TOOLS FOR ELECTRONIC PROOF CORRECTION

Required software to e-Annotate PDFs: Adobe Acrobat Professional or Adobe Reader (version 7.0 or above). (Note that this document uses screenshots from Adobe Reader X)
The latest version of Acrobat Reader can be downloaded for free at: http://get.adobe.com/uk/reader/

Once you have Acrobat Reader open on your computer, click on the Comment tab at the right of the toolbar:



This will open up a panel down the right side of the document. The majority of tools you will use for annotating your proof will be in the Annotations section, pictured opposite. We've picked out some of these tools below:

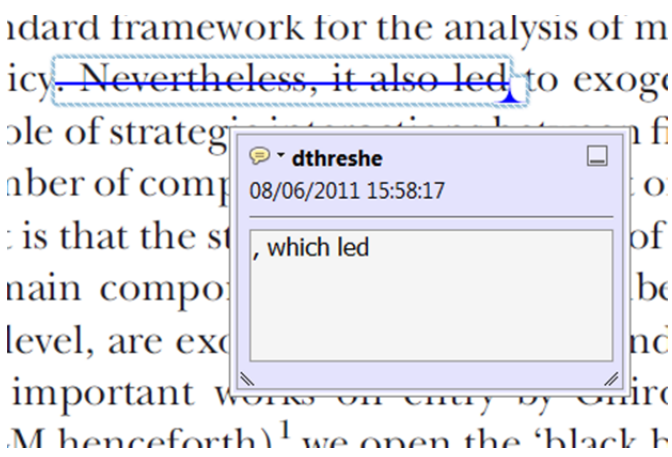


1. Replace (Ins) Tool – for replacing text.

Strikes a line through text and opens up a text box where replacement text can be entered.

How to use it

- Highlight a word or sentence.
- Click on the Replace (Ins) icon in the Annotations section.
- Type the replacement text into the blue box that appears.

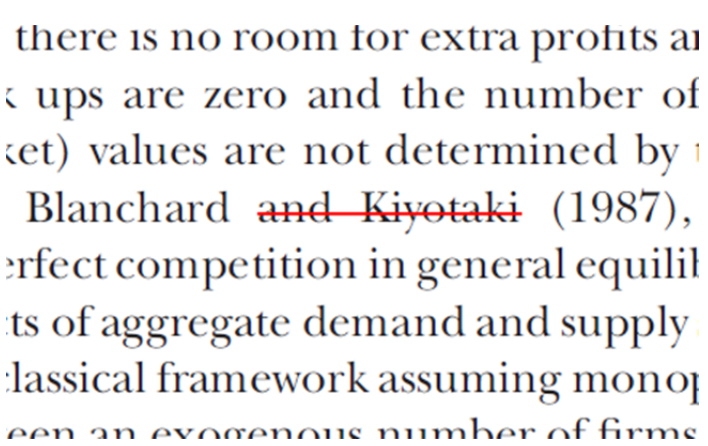


2. Strikethrough (Del) Tool – for deleting text.

Strikes a red line through text that is to be deleted.

How to use it

- Highlight a word or sentence.
- Click on the Strikethrough (Del) icon in the Annotations section.

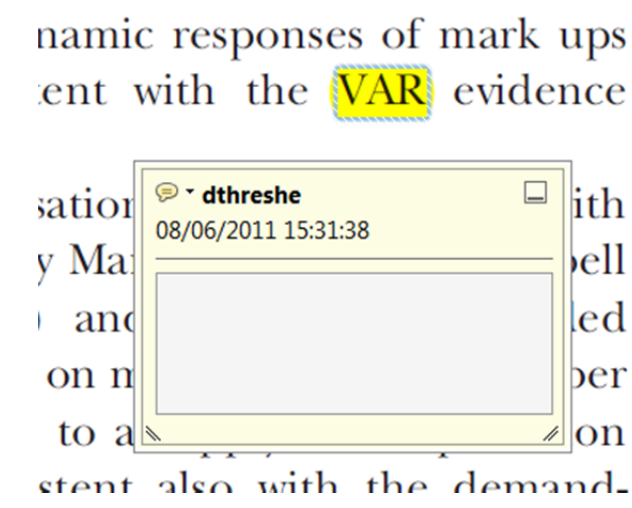


3. Add note to text Tool – for highlighting a section to be changed to bold or italic.

Highlights text in yellow and opens up a text box where comments can be entered.

How to use it

- Highlight the relevant section of text.
- Click on the Add note to text icon in the Annotations section.
- Type instruction on what should be changed regarding the text into the yellow box that appears.

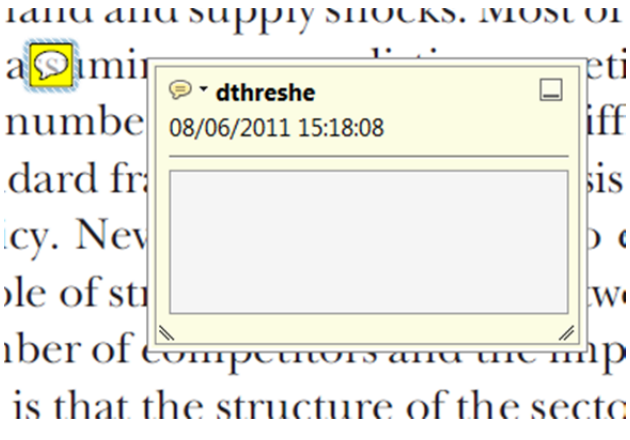


4. Add sticky note Tool – for making notes at specific points in the text.

Marks a point in the proof where a comment needs to be highlighted.

How to use it

- Click on the Add sticky note icon in the Annotations section.
- Click at the point in the proof where the comment should be inserted.
- Type the comment into the yellow box that appears.



WILEY

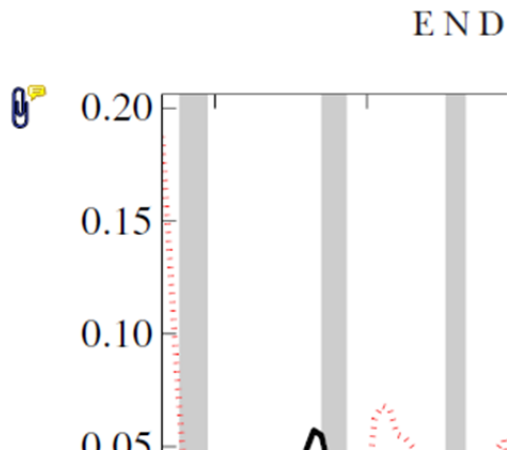
USING e-ANNOTATION TOOLS FOR ELECTRONIC PROOF CORRECTION

5. Attach File Tool – for inserting large amounts of text or replacement figures.

Inserts an icon linking to the attached file in the appropriate pace in the text.

How to use it

- Click on the Attach File icon in the Annotations section.
- Click on the proof to where you'd like the attached file to be linked.
- Select the file to be attached from your computer or network.
- Select the colour and type of icon that will appear in the proof. Click OK.

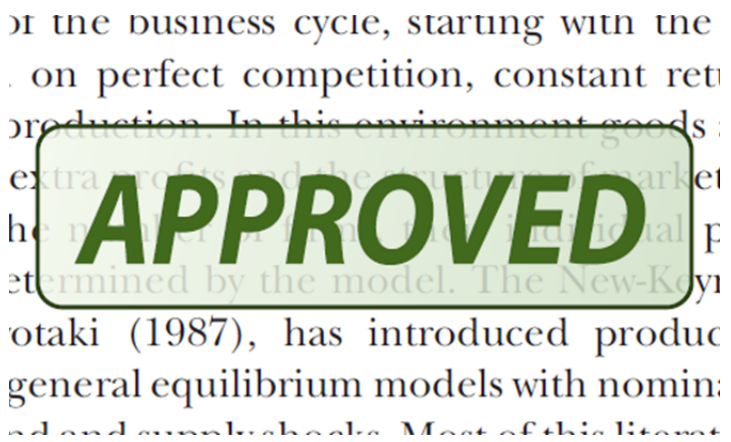


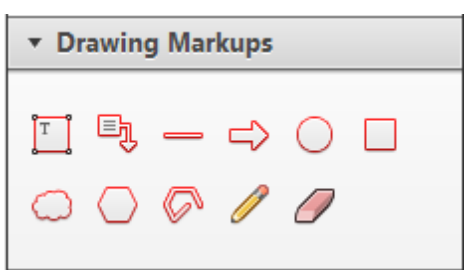
6. Add stamp Tool – for approving a proof if no corrections are required.

Inserts a selected stamp onto an appropriate place in the proof.

How to use it

- Click on the Add stamp icon in the Annotations section.
- Select the stamp you want to use. (The Approved stamp is usually available directly in the menu that appears).
- Click on the proof where you'd like the stamp to appear. (Where a proof is to be approved as it is, this would normally be on the first page).



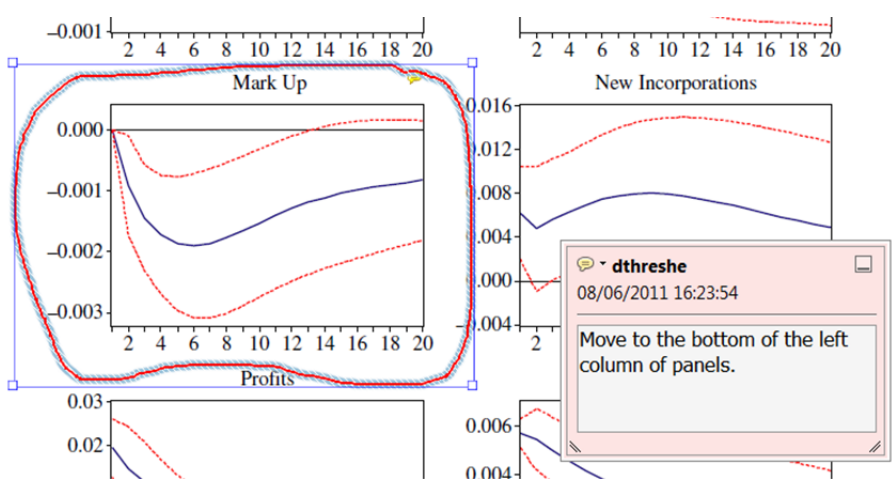


7. Drawing Markups Tools – for drawing shapes, lines and freeform annotations on proofs and commenting on these marks.

Allows shapes, lines and freeform annotations to be drawn on proofs and for comment to be made on these marks..

How to use it

- Click on one of the shapes in the Drawing Markups section.
- Click on the proof at the relevant point and draw the selected shape with the cursor.
- To add a comment to the drawn shape, move the cursor over the shape until an arrowhead appears.
- Double click on the shape and type any text in the red box that appears.



For further information on how to annotate proofs, click on the Help menu to reveal a list of further options: